A general framework for 3D model co-alignment

Xuanmeng Xie, Shan Luo, Jieqing Feng*

State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, Zhejiang, PR China

Abstract

3D model alignment is a fundamental step in many shape analysis processes, and various algorithms have been proposed to solve this problem. However, to the best of our knowledge, they are effective only on specific categories of models. Therefore, we present a novel framework that can align general categories by combining different features together. In order to align given groups of models, multiple features are evaluated first in this framework, according to three types of quantified characteristics, i.e., the intensity, the uniqueness and the consistency. Then, the quantified characteristics are combined into scores and a data-driven model is learned to predict the alignment errors according to the scores. Finally, the features with the minimum predicted alignment errors are selected to align the given groups. Experimental results show that our framework can generate consistent alignments on general categories, which are much better than those generated using single features.

Keywords: model alignment, co-alignment, feature evaluation, feature selection, shape analysis

1. Introduction

Model alignment is a fundamental step in many 3D shape analysis processes, such as shape matching, retrieval, recognition and other applications. In shape matching, models are aligned for comparison. In model retrieval and recognition, 3D shape descriptors play an important role, but many of them are not rotation invariant, and hence a preprocessing step of model alignment should be performed before the extraction of the descriptors. When organizing model galleries, the models should be aligned so that they are presented in consistent orientations.

The primary goal of model alignment is consistency. For example, in Figure 2(a), the heads of all the birds point outward from the paper and their torsos point rightward, so they are consistently aligned. In contrary, the 2nd one in Figure 2(b), the 3rd one in Figure 2(c), the 2nd one in Figure 2(d) and the 2nd one in Figure 2(e) are all inconsistently aligned with their corresponding groups.

However, model alignment is a challenging problem, because of the diversity of models. Many algorithms have been proposed to solve the problem. In one class of the algorithms, different features are employed for alignment, such as the slenderness, the reflective symmetry, the rotational symmetry, the rectilinearity, the minimum or maximum projection area, etc. In another class of algorithms, an orientation relevant function is defined to represent a model and then models are aligned by minimizing the distance between the functions. However, all these algorithms work well only on specific categories (a small set of categories) and they cannot be applied to general categories (almost all categories) directly.

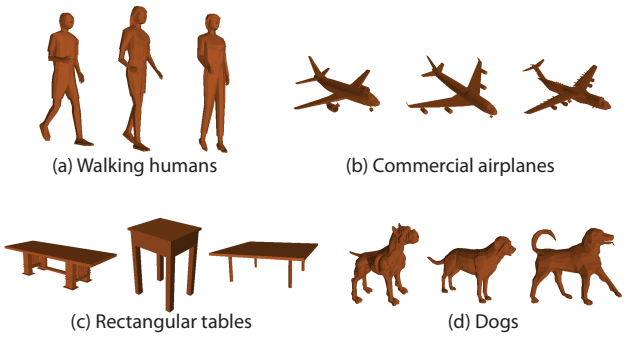

Figure 1: Four exemplar groups of models for alignment.

In order to reduce the "semantic gap" between 3D models and human understandings, our framework simulates the alignment procedure performed by human. According to our observations, when humans are faced with a group of models, one possible solution for aligning them is to look for the most significant common feature of them, and simultaneously consider whether the feature leads to unique and consistent alignments. For example, in Figure 1(a), humans may notice that the models are all slender, and thus employ the principal component analysis (PCA) to align them. Motivated by this process, we present an evaluation-based method of quantifying the extent of how a feature is suitable for aligning given groups of models, based on three types of criteria, i.e., the intensity, the uniqueness and the consistency.

*Corresponding author
Email address: jqfeng@cad.zju.edu.cn (Jieqing Feng)

Intensity. In Figure 1(a), we observe that the models are slender but not reflectively symmetric. In other words, they have high intensity of the slenderness and low intensity of the reflective symmetry, and, thus, the PCA is suitable for aligning them. By contrast, the PCA is not suitable for aligning the models shown in Figure 1(b), but the reflective symmetry method is.

Uniqueness. In Figure 1(c), the models are reflectively symmetric, but the reflective symmetry method cannot guarantee consistent alignments, because the planes of reflection are not unique.

Consistency. Most models shown in Figure 1(d) have high intensity of the reflective symmetry, except the last one, but the reflective symmetry method still works poorly on this group because of the inconsistency of the intensity values of the models.

The values of these criteria cannot be compared directly between different features, because the computation methods are different, and hence it is still unknown which is the best feature to align given groups of models. To solve this problem, we define a score by combining the values of the criteria, and then learn a data-driven model which represents the relation between the alignment errors and the scores. Finally, the features with the minimum predicted alignment errors are selected to align the given models.

Because features are effective only on specific categories, using single features to align general categories results in poor alignments. By combining different features, the framework presented in this paper is able to align general categories, and the alignment results are significantly improved. Moreover, by applying different features to align the first and the secondary axes, respectively, our framework can generate new alignments which cannot be generated using single features. For example, in Figure 2, all the single features fail to align both the first and the secondary axes, but our framework is able to align them consistently.

In this paper, a general framework is presented for model alignment, in which multiple features are evaluated and combined to yield consistent alignment results. Specifically, the contributions are as follows: (1) we propose an evaluation method of quantifying the extent of how a feature is suitable for aligning given groups of models, based on the intensity, the uniqueness and the consistency; (2) a data-driven method is proposed to select the best features for alignment.

2. Related Work

The algorithms for model alignment can be classified into two categories, i.e., the groupwise alignment (co-alignment) and the pairwise alignment. In this section, we majorly review the co-alignment algorithms and the related features, which are the most related to our framework. For completeness, we also review the pairwise alignment algorithms at the end of this section.

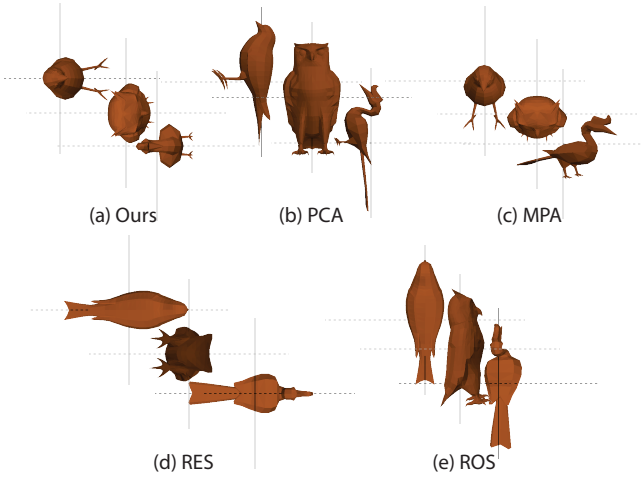

Figure 2: The alignments generated by our framework (Ours), the PCA (PCA), the maximum projection area (MPA), the reflective symmetry (RES) and the rotational symmetry (ROS). All the single features fail to align them, but our framework is able to align them consistently by selecting the RES and the MPA to align the first and the secondary axes, respectively. The solid lines represent the first axes and the dashed lines represent the secondary axes. Note that the MPA aligns models with the normals of the planes with the maximum projected areas, and the RES aligns models with the normals of the planes of the reflective symmetry.

For the groupwise alignment (e.g., [1, 2]), alignment is performed on the groups that contain models from similar categories. The models are generally aligned by employing one or multiple features. The best-known algorithms are the PCA-based algorithms, including the Continuous PCA (CPCA) [3] and the Normal-based PCA (NPCA) [4]. The CPCA method treats a mesh model as an infinite point set which consists of all the points on the surface of the model, whereas the NPCA method is applied on the face normals. The PCA-based algorithms are effective on the models that have different scales along different directions, especially on slender models, but they may generate inconsistent alignments on other types of models.

The symmetry, including the reflective symmetry and the rotational symmetry, is another major type of feature for model alignment, because the normals of the planes of reflection and the axes of rotation can be used to generate accurate alignments. The existing symmetry detection algorithms have two different goals: global symmetry detection [5, 6, 7, 8, 9, 10, 11, 12, 13] and partial symmetry detection [14, 15, 16]. The global symmetry detection techniques focus on finding symmetric planes or axes through the centers of the models; in contrast, the partial symmetry detection techniques consider all the possible planes or axes. In general, only the global algorithms are considered for model alignment. The partial algorithms can also detect global symmetries, but are less efficient.

The projection area is also widely used for model alignment. Johan *et al.* [17] proposed a projection-area-based algorithm, in which two perpendicular axes with the minimum projection areas are selected successively and then used for alignment. Lian *et al.* [18] proposed the concept of "rectilinearity" of 3D models, and models are aligned when the sums of the projection areas on the three coordinate planes are minimized.

Several features do not belong to the major types above, but are also effective in model alignment. Barequet and Har-Peled [19] proposed an efficient algorithm to compute the minimum-volume bounding boxes (MBB) of models, and the models are aligned based on the MBB axes and widths. Sfikas *et al.* [20] proposed an alignment algorithm based on panoramic views. In this algorithm, a plane of reflection is estimated by rotating the 3D model and computing the score measuring the extent of symmetry, and, thus, the first axis is estimated. The secondary axis is estimated by analyzing the variance of the model. In recent years, researchers propose several algorithms [21, 22, 23] to estimate the upright orientations of 3D models, and these orientations can also be used for partial alignment of one axis.

Observing that each type of feature is effective only on specific categories of models, several algorithms have been proposed to combine multiple features to enhance the alignment results. Chaouch and Verroust-Blondet [24] combine the CPCA with the symmetry. First, they use the CPCA to compute three axes, each with two opposite orientations, and then use symmetric characteristics to select the best axis and the best orientation simultaneously. Lian *et al.* [18] begin with the PCA and the rectilinearity to generate two types of alignments, respectively, and the alignments with less valid pixels of the silhouettes projected on three coordinate planes are finally selected. Axenopoulos *et al.* [25] proposed the Combined Pose Estimation method, in which the CPCA, the symmetry and the rectilinearity are combined. The CPCA is applied to yield the initial alignment. If reflective symmetry is observed in two or three CPCA-coordinate planes, the alignment is kept; otherwise, the rectilinearity is utilized to generate the final result. These combinational methods are able to generate better alignment results than does the use of single features, but they use simple criteria for feature selection, and the results could be further improved if more elaborate selection criteria are applied and more features are employed.

For the pairwise alignment, a common process is to represent a 3D model as an orientation relevant function, and a pair of models is aligned by finding the rotation that minimizes the distance between the functions. According to the types of the employed descriptors, these approaches can be further classified into two categories, i.e., the spatial-domain based algorithms [26, 27, 28] and the frequency-domain based algorithms [6, 29, 30]. The spatial-domain based algorithms are directly applied on the positions of the elements of the models. Whereas, in the frequency-domain based algorithms, the representations of the models are transformed from spatial domain to frequency domain, and the subsequent processes are performed in frequency domain. When the pairwise algorithms are used to align a large set of models, a reasonable strategy is to select a reference model and align other models to it; another strategy is to consider (but not truly align) all possible pairs and then remove or speed up unnecessary alignments with different techniques. The former strategy would generate poor alignments if the reference model is not properly selected, whereas the latter one results in lower efficiency compared with the former one.

3. Feature evaluation

In theory, an unlimited number of features can be registered into the framework. In order to select the best features for alignment, we employ three types of quantified characteristics to define the criteria for selection, i.e., the intensity, the uniqueness and the consistency.

To calculate the values of the criteria, a sampling strategy is adopted. Specifically, $N \times N$ axes are sampled in terms of the azimuthal and the elevational angles, where N is a positive integer. The range of the azimuthal angle is $[0, \pi)$ and the range of the elevational angle is $[-\pi/2, \pi/2)$. Because the axes for alignment are undirected, this configuration covers all possible axes. This sampling strategy is non-uniform, but it is easy to use and is enough for our purpose. Other reasonable sampling strategies (uniform or non-uniform) could also be considered.

In our implementation, N is set to 128 (it do not need to be a power of 2). Typically, a larger N can bring higher alignment accuracy, but it reduces the computational efficiency. Empirically, this value is recommended to be set between 30 and 300.

In the following, a detailed description of these criteria will be provided and the extraction procedure will be demonstrated with four exemplar features, i.e., the slenderness, the maximal projection area, the reflective symmetry and the rotational symmetry.

3.1. Intensity

The intensity measures the significance of features and it is the most important among the three types of criteria.

The slenderness can be measured by the PCA, and the intensity value is negatively correlated to the ratio of the second largest eigenvalue to the largest eigenvalue. But the PCA provides only three eigenvalues, which are not sufficient to measure all the sample axes. Thus, we come back to the nature of the PCA: the PCA is the optimal linear transformation which divides a space into orthogonal subspaces with the largest variance. Therefore, for a mesh model with M faces and a sample axis, a_i and $\boldsymbol{p}_i$ being the area and center of a face indexed by i, respectively, we measure the intensity value of the slenderness $I_s^{'}$ using the area-weighted variance:

$$I_s^{'} = \sqrt{\frac{\sum_{i=0}^{M} a_i((\boldsymbol{p}_i - \boldsymbol{c}) \cdot \boldsymbol{o})^2}{\sum_{i=0}^{M} a_i}}$$

where $\boldsymbol{c}$ is the center of the model and $\boldsymbol{o}$ is a unit vector along the axis.

The intensity of the maximum projection area I_p is calculated as:

$$I_p = \frac{\sum_{i=0}^{M} a_i \left| \boldsymbol{n}_i \cdot \boldsymbol{o} \right|}{\sum_{i=0}^{M} a_i}$$

where $\boldsymbol{n}_i$ is the normal of a face indexed by i.

The algorithm proposed by Kazhdan *et al.* [7] provides quantified measurements of both the reflective and rotational symmetries simultaneously. Suppose the values measuring the reflective and rotational symmetries along an axis are v_{re} and v_{ro}, respectively. The intensity of the reflective symmetry I_{re} is:

$$I_{re} = \begin{cases} 0, & v_{re} < v_{re,l} \\ \dfrac{\ln v_{re} - \ln v_{re,l}}{\ln v_{re,u} - \ln v_{re,l}}, & v_{re,l} \le v_{re} \le v_{re,u} \\ 1, & v_{re} > v_{re,u} \end{cases}$$

where $v_{re,l}$ and $v_{re,u}$ are the manually selected lower and upper bounds of v_{re}, respectively.

Similarly, the intensity of the rotational symmetry I_{ro} is:

$$I_{ro} = \begin{cases} 0, & v_{ro} < v_{ro,l} \\ \dfrac{\ln v_{ro} - \ln v_{ro,l}}{\ln v_{ro,u} - \ln v_{ro,l}}, & v_{ro,l} \le v_{ro} \le v_{ro,u} \\ 1, & v_{ro} > v_{ro,u} \end{cases}$$

where $v_{ro,l}$ and $v_{ro,u}$ are the manually selected lower and upper bounds of v_{ro}, respectively.

Note that the manually selected bounds are not real bounds, and they are selected according to the distributions of the values measuring the symmetries. For the symmetry estimation algorithm that we adopt, the real lower and upper bounds are 0 and infinity respectively, for both v_{re} and v_{ro}, which cannot be applied in the expressions to calculate the intensity values. The range between the manually selected upper and lower bounds should cover most values (e.g., $> 95\%$), and just abandon a few values which are too large or two small. In our implementation, $[2, 128]$ and $[2.5, 1000]$ are selected as the range for the reflective symmetry and the rotational symmetry, respectively (the voxelization resolution of the symmetry estimation algorithm is set to 128).

For convenience, the intensity values should be normalized in $[0, 1]$. The intensity values of the maximum projection area, the reflective symmetry and the rotational symmetry are already normalized according to the calculation expressions of them. In order to normalize the intensity value of the slenderness, we find the one with the largest intensity value $I_s^{''}$ from all the sample axes that are perpendicular to the axis corresponding to $I_s^{'}$. Considering the three eigenvalues of the PCA, $I_s^{'}$ and $I_s^{''}$ can be compared to the largest and the middle eigenvalues, respectively. The normalized intensity value of the slenderness is:

$$I_s = 1 - \frac{I_s^{''}}{I_s^{'}}$$

After the calculation of the intensity values related to all the sample axes, for each feature, the largest intensity value is selected to be the intensity value of the whole model, and the corresponding axis is selected to be the axis for alignment.

3.2. Uniqueness

If there are multiple similar intensity values that correspond to different sample axes, the alignment would be ambiguous. For this reason, the criterion of the uniqueness is defined to measure the uniqueness of the largest intensity value.

The uniqueness value of a feature is calculated based on the intensity values related to all the sample axes. Because of the spatial continuity, the differences of the intensity values between neighboring axes are typically small. Therefore, only the local maxima are considered, and an intensity value is considered as a local maximum if it is the largest in its K-degree neighborhood. Specifically, one axis is considered to be inside another's neighborhood if the angle between them is less than K degree. Then, for all the four types of exemplar features, the uniqueness value U of a model is defined as the ratio of the second largest local maximum I_{second} to the largest local maximum I_{max}.

$$U = 1 - \frac{I_{second}}{I_{max}}$$

Hence, the range of the uniqueness is $[0, 1]$. If there is only one local maximum, the uniqueness value is set to 1.

K should be properly set. If it is too large, reasonable local maxima might be covered by others' neighborhood. If it is too small, the second largest local maximum would generally appear near the largest one with the similar values, and the calculated uniqueness value would be not suitable to measure whether there are ambiguous axes with similar intensity values. In our implementation, K is set to 10.

3.3. Consistency

The criterion of the consistency measures how consistent a model is with other models in the same group. The

consistency can be reflected in the values of the intensity and the uniqueness. Hence, two types of consistency are defined, i.e., the consistency of the intensity and the consistency of the uniqueness. For a model, the consistency value of the intensity is defined as the absolute value of the difference between its intensity value and the average intensity value of the corresponding group, and the consistency value of the uniqueness is calculated similarly.

$$C_I = 1 - |I - I_{avg}|$$

$$C_U = 1 - |U - U_{avg}|$$

In these equations, C_I and C_U are the consistency value of the intensity and the consistency value of the uniqueness, respectively; I_{avg} and U_{avg} are the average intensity value and the average uniqueness value of the group, respectively.

4. Feature selection

In our framework, model alignment is divided into two steps, i.e., the estimation of the first coordinate axes and the estimation of the secondary ones. Then, the third axes are the cross products of them. After deploying the reference frames, models are naturally aligned. In both steps, we apply the same method to select the best features for specific partial alignments, except that the axes are sampled only in the planes that are perpendicular to the first axes when estimating the secondary axes. Thus, only the procedure for estimating one type of axes will be demonstrated.

For each type of axes, the estimation procedure contains a training phase and a test phase, and the details are demonstrated in Figure 3 and Figure 4, respectively. To avoid cluttering in the figures, only two features are demonstrated. The overall estimation procedure is as follows. First, we define a score for each feature based on the criterion values and the parameters of the score calculation expression are optimized by minimizing a defined energy. Second, for each feature, a curve is fitted to represent the relation between the scores and the angular errors. Finally, with the score calculation expressions and the curves, the feature with the minimum average predicted angular error is selected for alignment.

4.1. Score definition

In Section 3, we have described the criteria for evaluating the extent of the suitability of a feature in aligning a group of models, and the values of these criteria should be transformed into a single score for convenience of comparison.

The score is defined to measure the ability of a feature to align given models. The intensity can do the same thing, but it does not work well alone. Therefore, the uniqueness, the consistency of the intensity and the consistency of the uniqueness are combined with the intensity one by one to enhance the power of the score.

Suppose f is a function with four inputs: u, v, $para$ and op, where $op \in \{1, 2\}$ is a parameter determining the operation of f, and $para \in [0, \infty)$ is a parameter controlling the extent of the effect of v on u.

$$f(u, v, para, op) = \begin{cases} u + v \cdot para, & op = 1 \\ u \cdot v^{para}, & op = 2 \end{cases}$$

The score s is defined as:

$$s_0 = I$$

$$s_1 = f(s_0, U, para_1, op_1)$$

$$s_2 = f(s_1, C_I, para_2, op_2)$$

$$s = f(s_2, C_U, para_3, op_3)$$

These expressions can be explained as follows: The score is originally equal to I. Then, U, C_I, and C_U take effects on the value of the score in turn. The parameters of *op*s determine how they take effects and the parameters of *para*s determine the extents of the effects. Finally, the score is the combination of the intensity I, the uniqueness U, the consistency of the intensity C_I and the consistency of the uniqueness C_U with the related parameters.

The parameters of the expressions are optimized by energy minimization, which will be explained in Section 4.3. The score is defined in a cascaded way so that we can optimize one pair of parameters independently in a cascade.

4.2. Energy definition

Because an unlimited number of features can be registered into our framework in theory, it is not practical to manually adjust the parameters for each feature. Therefore, we define an energy function and the parameters are optimized by minimizing the energy.

In the training set, there are plenty groups of models, which are aligned manually. These alignments serve as the ground-truth and the angular errors of other alignments are computed through comparison with them.

For a model indexed by i, suppose the angles between the estimated axis (the first or the secondary axis) and the three manually aligned axes are $A_{x,i}$, $A_{y,i}$ and $A_{z,i}$, respectively. The average angles of the group are $A_{x,avg}$, $A_{y,avg}$ and $A_{z,avg}$, respectively. The angular error e_i of the model with respect to the group is defined as:

$$e_i = \frac{|A_{x,i} - A_{x,avg}| + |A_{y,i} - A_{y,avg}| + |A_{z,i} - A_{z,avg}|}{3}$$

The definition of the angular error is explained as follows. As demonstrated in Figure 5, the alignments of (a) and (b) are appropriately the same in nature, but they are represented with different axes. It is similar to the

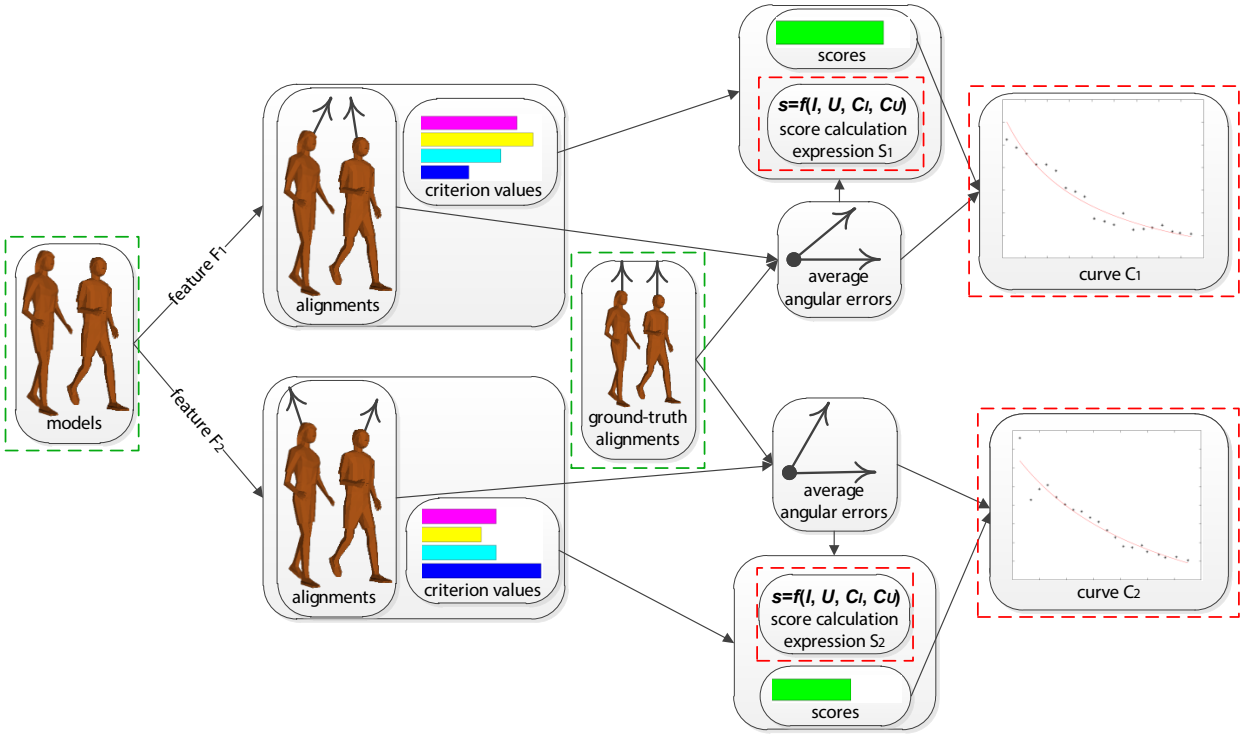

Figure 3: The procedure for the training phase of our framework with respect to one type of axes (the first or the secondary axes). First, by applying different features, we yield several groups of alignments together with four types of criterion values, i.e., the intensity, the uniqueness, the consistency of the intensity and the consistency of the uniqueness. Second, the average angular errors are computed through comparison between the produced alignments and the ground-truth. Third, the parameters of the score calculation expressions are learned by energy minimization and the scores are computed simultaneously. Finally, for each feature, a curve is fitted to represent to relation between the scores and the average angular errors. The inputs are framed in green, and the outputs are framed in red.

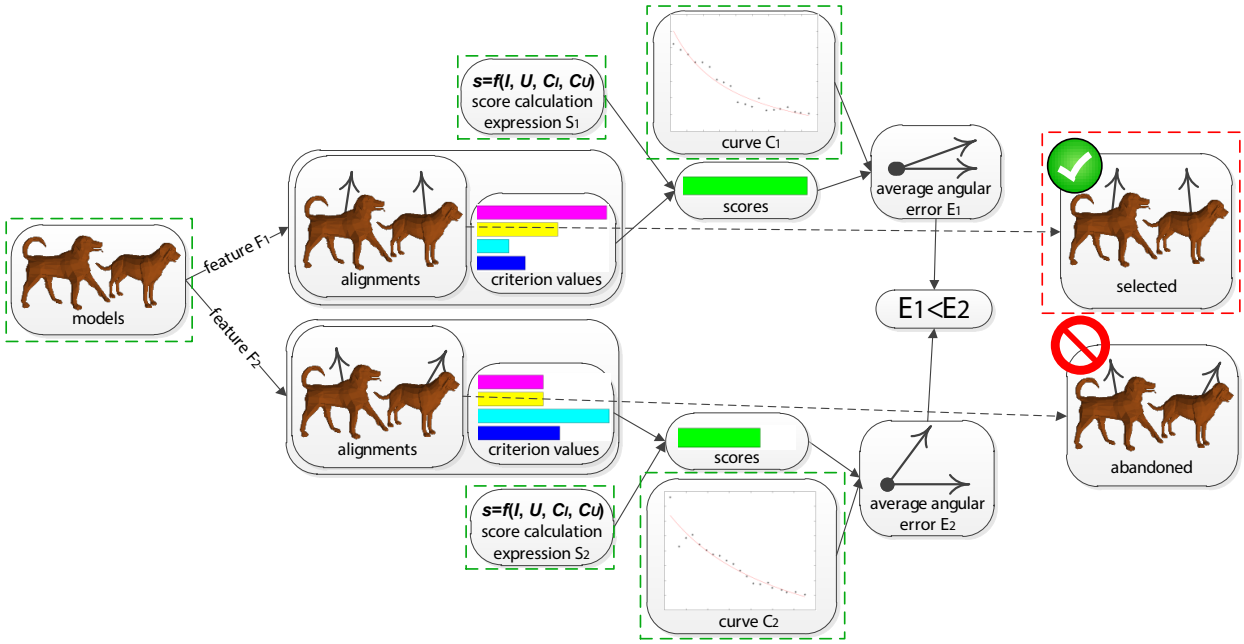

Figure 4: The procedure for the test phase of our framework with respect to one type of axes. First, different features are applied to yield several groups of alignments together with four types of criterion values. Second, the scores are computed based on the criterion values and the score calculation expressions. Third, the average angular errors are predicted with the scores and the curves that are learned in the training phase. Finally, the alignments with the minimum average predicted angular error are selected as the final results. The inputs are framed in green, and the outputs are framed in red.

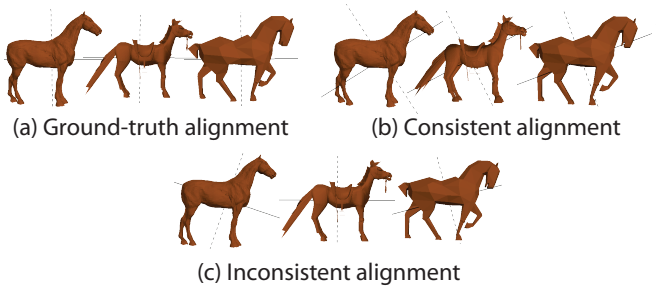

Figure 5: Alignments of the same group, which are represented by the first (solid lines) and the secondary axes (dashed lines).

situation that the same point may have different coordinates with regard to different reference frames. Therefore, the angular error cannot be defined directly with the angles between corresponding axes, because the angular error cannot measure the extent of alignment consistency. We observe that $A_{x,i}$ is appropriately a constant for the models in (b), and, hence, $|A_{x,i} - A_{x,avg}|$ is appropriately equal to 0, but $A_{x,i}$ differs greatly for the models in (c), which are inconsistently aligned. This observation also holds for $A_{y,i}$ and $A_{z,i}$. Therefore, $|A_{x,i} - A_{x,avg}|$, $|A_{y,i} - A_{y,avg}|$ and $|A_{z,i} - A_{z,avg}|$ can be used for estimating the consistency of an alignment, and the expression above is adopted to define the angular error.

Then the energy E of the group is defined as:

$$E = \frac{\sum_i \sum_j sgn\,(s_i - s_j) \cdot (e_i - e_j)}{\sum_i \sum_j |e_i - e_j|}$$

where s_i and s_j are the scores of models indexed by i and j, respectively. Here, sgn is the signal function such that:

$$sgn\,(x) = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$$

Ideally, a larger score corresponds to a lower angular error. In this case, $sgn\,(s_i - s_j) \cdot (e_i - e_j)$ is a negative value. In the next step, we adjust the parameters of the score calculation expressions so as to minimize the energy. By adopting this definition of energy, minimizing the energy is equivalent to improving the quantity of ideal cases.

4.3. Energy minimization

A greedy algorithm is applied to learn the parameters of the score calculation expressions. First, the parameters of *op*s are set to 1, and the parameters of *para*s are set to 0. Then, the parameters are optimized iteratively. In each iteration, only one pair of parameters is optimized to minimize the energy. The process goes on until the energy stops decreasing and it typically terminates in less than 10 iterations. Empirically, it is sufficient to obtain good minimization results by exhaustively testing the values of the parameters of *para*s in this set:

$$\{0\} \cup \{1.1^n | n \in N, n \in [-50, 50]\}$$

Finally, the values that minimize the energy are assigned to the parameters of the score calculation expressions. The details are shown in Algorithm 1.

4.4. Curve fitting

The criterion values related to different features are calculated with different methods, thereby having different intrinsic meanings. Because the scores rely on the criterion values, they cannot be employed for feature selection directly. By fitting curves that represent the relation between the scores and the angular errors, we are able to predict the angular errors with the scores and, consequently, the feature with the minimum predicted angular error can be selected to align the models. According to the relation between the scores and the angular errors, we employ inverse proportion curves to fit them in our implementation. Because the angular errors predicted by different features often differ greatly, we do not require high accuracy of the fitted curves, and inverse proportion curves work well in our experiments. However, other reasonable curves could also be considered. The expression of an inverse proportion curve is defined as:

$$y = \frac{1}{b_1 \cdot x + b_2} + b_3$$

where b_1, b_2 and b_3 are three parameters controlling the shape and the position of the curve.

When fitting the curves, each pair of the score and the angular error corresponds to a point in the chart, and these points are fitted using least squares method. Because the data are noisy, they are not directly used for fitting. Instead, the range of the scores is equally divided into k parts, and the average values of these parts are employed for fitting. This treatment markedly improves the fitting accuracy and the fitting results will be shown in Section 5.2.

5. Experimental Results

In our experiments, four types of features were employed to evaluate the effectiveness of our framework, i.e., the slenderness, the maximum projection area, the reflective symmetry and the rotational symmetry. For simplicity, we define the abbreviations for them, as shown in Table 1. All the experiments were performed on a laptop equipped with an Intel i7-3630QM CPU at 2.4 GHz (only one core was used).

5.1. Dataset

Our framework was tested on the dataset from the Princeton Shape Benchmark [31], which consists of 1814 3D polygonal models. The models in the dataset have already been divided into a training set and a test set,

Algorithm 1 Energy minimization and parameter optimization

Input: $inputs$ //all the inputs are used by the energy calculation function
Output: The parameters op_1, $para_1$, op_2, $para_2$, op_3, $para_3$; The minimum energy E_{min};

Declare the energy calculation function: $E \leftarrow calE\,(op_1, para_1, op_2, para_2, op_3, para_3, inputs)$;
Define sets: $S_{op} \leftarrow \{1, 2\}$; $S_{para} \leftarrow \{0\} \cup \{1.1^n | n \in N, n \in [-50, 50]\}$;
$op_1 \leftarrow 1$; $op_2 \leftarrow 1$; $op_3 \leftarrow 1$; $para_1 \leftarrow 0$; $para_2 \leftarrow 0$; $para_3 \leftarrow 0$;
$ite \leftarrow 0$; $E_{min} \leftarrow \infty$;
repeat
  $index \leftarrow mod(ite, 3)$;
  if $index = 0$ **then** //optimize op_1 and $para_1$
    search op in S_{op} and $para$ in S_{para} to compute $E \leftarrow calE(op, para, op_2, para_2, op_3, para_3, inputs)$, and find op_{tmp} and $para_{tmp}$ with the minimum energy of E_{tmp};
    if $E_{tmp} < E_{min}$ **then**
      $E_{min} \leftarrow E_{tmp}$; $op_1 \leftarrow op_{tmp}$; $para_1 \leftarrow para_{tmp}$;
    end if
  else if $index = 1$ **then** //optimize op_2 and $para_2$
    search op in S_{op} and $para$ in S_{para} to compute $E \leftarrow calE(op_1, para_1, op, para, op_3, para_3, inputs)$, and find op_{tmp} and $para_{tmp}$ with the minimum energy of E_{tmp};
    if $E_{tmp} < E_{min}$ **then**
      $E_{min} \leftarrow E_{tmp}$; $op_2 \leftarrow op_{tmp}$; $para_2 \leftarrow para_{tmp}$;
    end if
  else //optimize op_3 and $para_3$
    search op in S_{op} and $para$ in S_{para} to compute $E \leftarrow calE(op_1, para_1, op_2, para_2, op, para, inputs)$, and find op_{tmp} and $para_{tmp}$ with the minimum energy of E_{tmp};
    if $E_{tmp} < E_{min}$ **then**
      $E_{min} \leftarrow E_{tmp}$; $op_3 \leftarrow op_{tmp}$; $para_3 \leftarrow para_{tmp}$;
    end if
  end if
  $ite++$;
until E_{min} does not change for 2 successive iterations

each containing 907 models. The benchmark also provides several classifications according to the models' categories, and the most detailed one is employed in our experiments. In this classification, the training set is classified into 90 groups, and the test set is classified into 92 groups.

In order to produce ground-truth alignments for comparison, the models in the dataset are manually aligned such that the models in the same groups have consistent orientations. Specifically, each model is assigned two perpendicular axes, which define a reference frame, and then the models in the same groups are aligned according to the reference frames.

Table 1: Abbreviations of Features

Feature	Abbreviation	References
Slenderness (evaluated by the PCA)	PCA	[3]
Maximum projection area	MPA	[18, 17]
Reflective symmetry	RES	[7]
Rotational symmetry	ROS	[7]

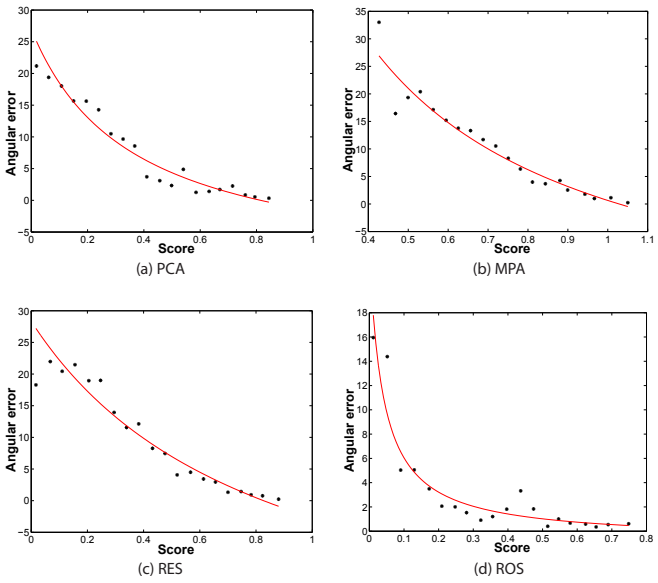

Figure 6: The fitted curves corresponding to the four employed features related to the first axes. The angular errors are shown in degrees.

5.2. Curve fitting

In our implementation, the number of parts k that the range of the scores is divided into was set to 20, and hence there were 20 points to be fitted in the chart if no part was empty. The fitting results related to the first axes are demonstrated in Figure 6. The results related to the secondary axes are similar and thus omitted.

We can see that, for each feature, the curve fits the data well except for three cases. (1) For the parts on the very left in Figure 6(b and c), the scores are relatively small, which indicates that the features probably fail to align the given models and, to some extent, the angular errors are random. (2) For the parts on the very right in Figure 6(a, b and c), the angular errors predicted by the curves may be less than 0, but that has little negative effect on the final alignments, because the selected features are still effective. (3) In the part with the a score from 0.3 to 0.5 in Figure 6(d), the curve does not fit the data well, because few original points fall into this range. (Note that the points shown in the chart are the average values of the original sample points in these parts, and average values are known to be inaccurate with insufficient number of samples.)

5.3. Feature selection

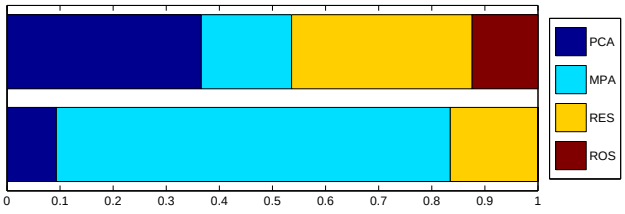

Figure 7: The results of the feature selection related to the first (the upper bar) and the secondary axes (the lower bar). The lengths of the bars represent the frequencies of the features being selected by the framework for alignment.

In this experiment, we show the results of feature selection on the test set, in both steps of aligning the first and the secondary ones. In Figure 7, the lengths of the bars represent the frequencies of the features being selected by the framework for alignment. We can see that the slenderness and the reflective symmetry were the most frequently selected features in aligning the first axes, whereas the maximum projection area was the most frequently selected feature in aligning the secondary axes. The rotational symmetry was never selected in aligning the secondary axes, because no significant rotational symmetry was detected in this step. In addition, 77.0% of the models in the test set were aligned with two different features in the two steps, respectively, and these alignments could not be generated using single features.

5.4. Computational time

In this experiment, we analyze the computational time of our framework. In terms of computational time, the training phase can be divided into three major steps, i.e., feature extraction, uniqueness computation and parameter learning; the test phase can be divided similarly except that it does not have the step of parameter learning. The time consumed by other steps is quite short and hence can be ignored. The detailed times of these major steps are shown in Table 2.

We can see that the framework took 6.43 seconds in average to train a model and took 7.55 seconds in average to align a model. The time for aligning a model was longer than that for training because the models in the test set are more complicated than those in the training set. Specifically, the models have 4373 vertices and 7960 faces in average in the test set, whereas the models have 4070 vertices and 7325 faces in average in the training set.

In traditional learning-based applications (e.g., [32]), the learning step generally costs much more time than that for feature extraction. In contrast, the learning step of our framework costs much less time, because our framework does not learn the alignments directly. Instead, it just learns the relationship between the angular errors and 4 parameters, i.e., the intensity, the uniqueness, the consistency of the intensity and the consistency of the uniqueness. Thus, the computational complexity is significantly reduced. In our experiment, the learning step cost 0.35 seconds in average for a model, and it comprised just 4.68% of the whole training time.

Table 2: Average Computational Times per Model (in seconds)

Step	Feature	Training Phase	Test Phase
Feature Extraction	PCA	1.26	1.42
	MPA	0.92	1.11
	RES	1.42	1.46
	ROS	1.73	1.65
Uniqueness Computation	PCA	0.90	0.92
	MPA	0.65	0.71
	RES	0.08	0.08
	ROS	0.19	0.21
Parameter Learning	×	0.35	×

5.5. Comparison with single features

Our framework was compared with the uses of single features in terms of alignment error. Because some applications only require the alignments of the first axes, whereas others require full alignments, the comparison was performed with two criteria, i.e., the average angular error related to the first axes and the average angular error related to all the three axes. The average angular errors

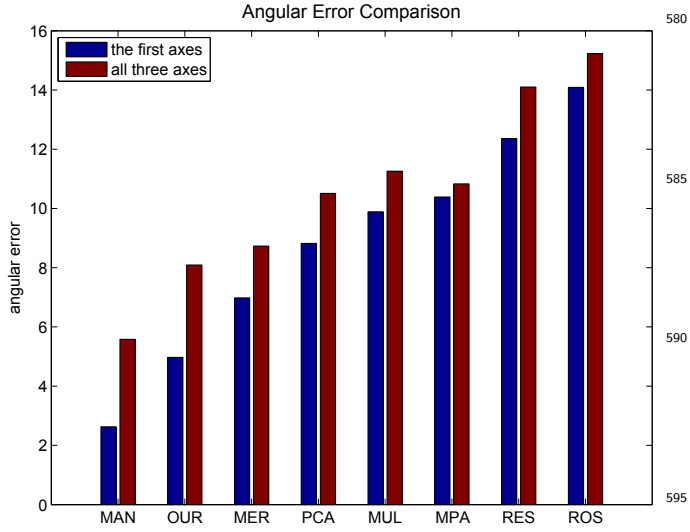

Figure 8: The comparison of the angular errors (in degrees). The blue bars represent the average angular errors related to the first axes, whereas the red bars represent the average angular errors related to all the three axes. MAN=manual alignments, OUR=our framework, MER=our framework with merged categories, MUL=multiple features (Axenopoulos *et al.*'s algorithm).

were calculated through comparison between the estimated alignments and the ground-truth alignments, using the same calculation method defined in Section 4.2.

The results of the comparison are shown in Figure 8. In addition, we show the angular errors of another group of manual alignments (MAN) in this figure, which is generated by a different person. Similar to the ground-truth alignments, this group of manual alignments has also been carefully checked so as to avoid observable mistakes. Because different persons may have different tendencies to align the same models, both groups of alignments are correct but not similar. This group of alignments gives an estimation of the limit of the angular errors that an algorithm can achieve in practice.

The results show that the angular errors of our alignments are much smaller than those generated using single features, demonstrating the effectiveness of our framework.

5.6. Comparison with Axenopoulos et al.*'s algorithm*

Because our framework employs multiple features, we also compare it with Axenopoulos *et al.*'s algorithm [25], which combines the CPCA, the symmetry and the rectilinearity. In this algorithm, the CPCA is applied to generate an initial alignment for each model. If reflective symmetry is observed in two or more CPCA-coordinate planes, the initial alignment is kept; otherwise, the model is realigned with the rectilinearity. This algorithm is a traditional normalization algorithm instead of a co-alignment algorithm, so it may use different features to align models in the same groups, and, consequently, generates poor alignments.

Therefore, we made some modifications to turn this algorithm into a co-alignment algorithm. First, each model was initially aligned with the CPCA. Then, for each group of models, we calculated the average scores that estimated the extents of the reflective symmetry with respect to the CPCA-coordinate planes. Finally, for a category, if two or more average scores were higher than a given threshold, the initial alignments were kept; otherwise, the rectilinearity was applied to generate new alignments.

Compared to our framework, the modified algorithm has the advantage of efficiency, because it extracts fewer features and does not need a training phase. Specifically, it took 2.29 seconds in average to align a model in the test set, whereas our framework took 7.55 seconds. However, the alignment results generated by our framework were much better. The angular errors related to the modified algorithm are also shown in Figure 8 (MUL). The reason why our framework performs better is that it has a more elaborate mechanism to select suitable features for aligning specific groups of models.

5.7. Comparison with Averkiou et al.*'s algorithm*

The descriptor-based algorithms play an important role in aligning 3D models, in which models are generally aligned pairwise by minimizing the distance between the corresponding descriptors. When they are applied to align a large set of models, a common strategy is to select a reference model and then align all the other models to it, but this strategy may generate poor alignments if the reference model is not properly selected. Instead, Averkiou *et al.*'s algorithm [26] considers all possible pairs of models and uses a divide-and-conquer strategy to speed up the computation. First, it divides the whole set into subsets such that the models in each subset are easier to be aligned; then, it aligns the subsets and, consequently, all the models are co-aligned.

Averkiou *et al.*'s algorithm supposes that the first axes have already been aligned and only aligns the secondary axes. Therefore, we used the ground-truth first axes for both Averkiou *et al.*'s algorithm and our framework and only compared the angular errors related to the secondary axes. The angular errors related to our framework and Averkiou *et al.*'s algorithm were 7.44 and 10.53 degrees, respectively, and the average times to align a model were 2.38 and 9.99 seconds, respectively. The results show that our framework achieves higher alignment accuracy with much less computational time.

5.8. Alignment results

The alignment results generated by our framework are shown in Figure 9. The estimated first axes are represented with solid lines, and the estimated secondary axes are represented with dashed lines. We can see that the framework can generate consistent alignments on general categories of models. Moreover, suitable features can be selected by the framework in each step, thereby guaranteeing the effectiveness of the framework.

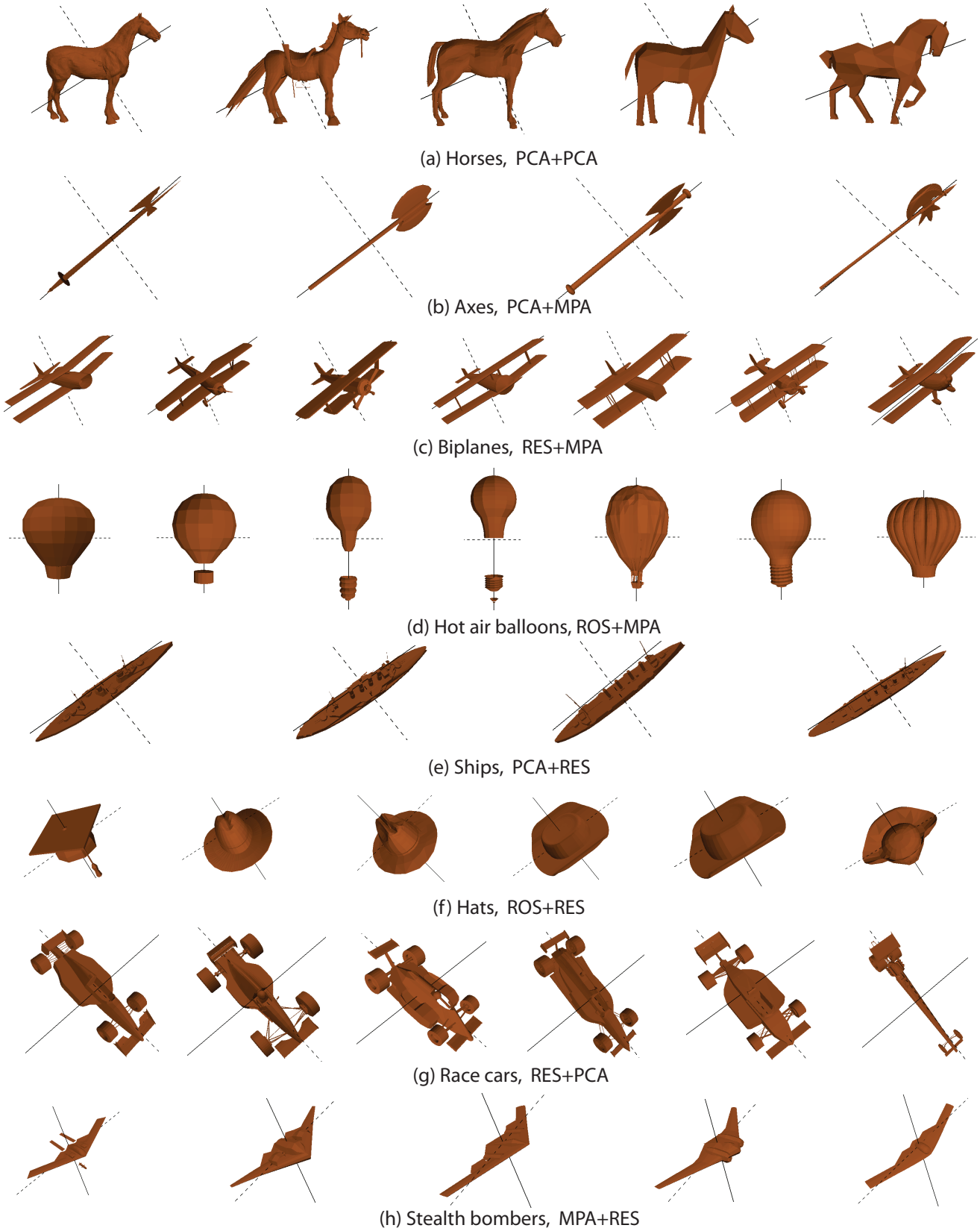

Figure 9: The alignment results generated by our framework. The solid lines represent the estimated first axes and the dashed lines represent the estimated secondary axes. The caption under each group shows the name of the category and the features selected by the framework to align the first and the secondary axes, respectively.

5.9. Inter-class alignment

In general, co-alignment algorithms should be applied on similar models, and the categorization of models provides a natural way to classify similar models into groups. Aligning dissimilar models (e.g., aligning a door to a bird or aligning a plane to a hat) is often meaningless, and humans are not aware how to align them.

In this experiment, we merged the 92 categories in the test set into 18 groups according to their similarities. With the new classification, most groups contained multiple categories. For example, balloon vehicles, heads of humans and wheels were merged; sea vessels, cars and cycles were merged; arthropod animals, humans and quadruped animals were merged.

With the new classification, the average angular errors related to the first axes and all three axes were 6.98 and 8.73 degrees, respectively, as shown in Figure 8 (MER). Compared to the alignment results with the original classification, the consistency of the alignments had a modest decrease, because the models in the new groups differed greatly in shape, which prevented our framework from selecting the most suitable features for alignment. However, the consistency was still higher than that of any single feature. This experiment proves that our framework can still work well on inter-class groups without accurate categorization of the models.

5.10. Failure cases

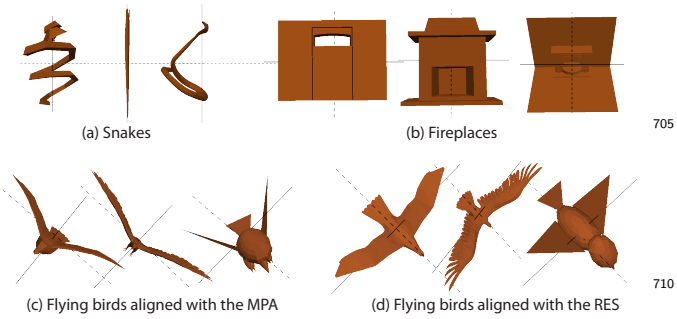

Figure 10: (a, b and c) show three failure cases of our framework. (d) shows the correct alignments generated with the RES.

Although our framework works well on most categories, there are several failure modes that result in inconsistent alignments.

First, as shown in Figure 10(a), the models of snakes vary greatly in shape and even humans are confused how to align them.

Second, as shown In Figure 10(b), although our framework selected the RES to align the first axes consistently, all the features failed to align the secondary axes, and, hence, the final alignments were not consistent.

Finally, as shown in Figure 10(c and d), our framework selected the MPA to align both axes of the models, but the RES generated better alignments. It is two-fold why the RES was not selected. On one hand, this group contained several thin models, and the reflective symmetry calculation algorithm [7] that we adopted might generate inaccurate values on thin models, which estimated the extents of the reflective symmetry. Although we can see that these thin models have unique planes of the reflective symmetry, this algorithm assigned similar values to two different planes for a single model. Thus, the RES got a relatively low value of the uniqueness, which prevented it from being selected by the framework. This problem could be solved by promoting the voxelization resolution of the adopted algorithm, or by adopting a new algorithm which performs better on thin models. On the other hand, although the values of the reflective symmetry were similar, they were still distinguishable. This problem might also be solved by adjusting the method of uniqueness calculation.

6. Conclusions and future work

In this paper, we propose a general framework for model alignment. Based on three types of quantified characteristics, multiple features are evaluated and the suitable ones are selected to align given groups of models. This framework can be applied to general categories and it can yield alignments with much smaller angular errors than can single features. Moreover, the framework can generate new alignments by combining two different features to align the first and the secondary coordinate axes, respectively, which cannot be generated using single features.

The major limitation of our framework is that the alignment results rely on the employed features. If all the features fail to align given models consistently, our framework cannot guarantee consistent results, either.

In future work, we plan to employ and test more features in our framework. In addition, more evaluation criteria, different energy functions and different types of curves could also be considered, which may further improve the effectiveness of the framework.

Acknowledgments

The authors would like to thank the anonymous reviewers who gave valuable suggestions to improve the quality of the paper. This work was supported by the National Natural Science Foundation of China under Grant No. 61472349.

References

[1] Q.-X. Huang, H. Su, L. Guibas, Fine-grained semi-supervised labeling of large shape collections, ACM Transactions on Graphics 32 (6) (2013) 190:1–190:10. doi:10.1145/2508363.2508364.

[2] J.-H. Chen, L. G. Shapiro, Groupwise Pose Normalization for Craniofacial Applications, in: IEEE Workshop on Applications of Computer Vision, 2011, p. 248255. doi:10.1109/WACV.2011.5711510.

[3] D. V. Vranić, 3D Model Retrieval, Ph.D. thesis, University of Leipzig (2004).

[4] P. Papadakis, I. Pratikakis, S. Perantonis, T. Theoharis, Efficient 3D shape matching and retrieval using a concrete radialized spherical projection representation, Pattern Recognition 40 (9) (2007) 2437–2452. doi:10.1016/j.patcog.2006.12.026.

[5] M. Kazhdan, B. Chazelle, D. Dobkin, T. Funkhouser, S. Rusinkiewicz, A Reflective Symmetry Descriptor for 3D Models, Algorithmica 38 (1) (2004) 201225. doi:10.1007/s00453-003-1050-5.

[6] M. Kazhdan, An Approximate and Efficient Method for Optimal Rotation Alignment of 3D Models, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (7) (2007) 1221–1229. doi:10.1109/TPAMI.2007.1032.

[7] M. Kazhdan, T. Funkhouser, S. Rusinkiewicz, Symmetry descriptors and 3D shape matching, in: ACM SIGGRAPH symposium on Geometry processing, 2004, pp. 115–123. doi:10.1145/1057432.1057448.

[8] N. J. Mitra, M. Pauly, M. Wand, D. Ceylan, Symmetry in 3D Geometry: Extraction and Applications, Computer Graphics Forum 32 (6) (2013) 1–23. doi:10.1111/cgf.12010.

[9] J. D. Wolter, T. C. Woo, R. A. Volz, Optimal algorithms for symmetry detection in two and three dimensions, The Visual Computer 1 (1) (1985) 37–48. doi:10.1007/BF01901268.

[10] C. Sun, J. Sherrah, 3D symmetry detection using the extended Gaussian image, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (2) (1997) 164–168. doi:10.1109/34.574800.

[11] K. Sfikas, T. Theoharis, I. Pratikakis, ROSy+: 3D Object Pose Normalization Based on PCA and Reflective Object Symmetry with Application in 3D Object Retrieval, International Journal of Computer Vision 91 (3) (2011) 262–279. doi:10.1007/s11263-010-0395-x.

[12] I. Sipiran, R. Gregor, T. Schreck, Approximate Symmetry Detection in Partial 3D Meshes, Computer Graphics Forum 33 (7) (2014) 131–140. doi:10.1111/cgf.12481.

[13] S. Korman, R. Litman, S. Avidan, A. Bronstein, Probably Approximately Symmetric: Fast Rigid Symmetry Detection With Global Guarantees, Computer Graphics Forum 34 (1) (2015) 2–13. doi:10.1111/cgf.12454.

[14] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, T. Funkhouser, A planar-reflective symmetry transform for 3D shapes, ACM Transactions on Graphics 25 (3) (2006) 549–559. doi:10.1145/1141911.1141923.

[15] K. Xu, H. Zhang, W. Jiang, R. Dyer, Z. Cheng, L. Liu, B. Chen, Multi-scale partial intrinsic symmetry detection, ACM Transactions on Graphics 31 (6) (2012) 181:1–181:12. doi:10.1145/2366145.2366200.

[16] J. Kerber, M. Bokeloh, M. Wand, H.-P. Seidel, Scalable Symmetry Detection for Urban Scenes, Computer Graphics Forum 32 (1) (2013) 3–15. doi:10.1111/j.1467-8659.2012.03226.x.

[17] H. Johan, B. Li, Y. Wei, Iskandarsyah, 3D model alignment based on minimum projection area, The Visual Computer 27 (2011) 565–574. doi:10.1007/s00371-011-0590-y.

[18] Z. Lian, P. L. Rosin, X. Sun, Rectilinearity of 3D Meshes, International Journal of Computer Vision 89 (2) (2010) 130–151. doi:10.1007/s11263-009-0295-0.

[19] G. Barequet, S. Har-Peled, Efficiently Approximating the Minimum-Volume Bounding Box of a Point Set in Three Dimensions, Journal of Algorithms 38 (1) (2001) 91–109. doi:10.1006/jagm.2000.1127.

[20] K. Sfikas, T. Theoharis, I. Pratikakis, Pose normalization of 3D models via reflective symmetry on panoramic views, The Visual Computer 30 (11) (2014) 1261–1274. doi:10.1007/s00371-014-0935-4.

[21] H. Fu, D. Cohen-Or, G. Dror, A. Sheffer, Upright orientation of man-made objects, ACM Transactions on Graphics 27 (3) (2008) 42:1–42:8. doi:10.1145/1399504.1360641.

[22] C.-K. Lin, W.-K. Tai, Automatic upright orientation and good view recognition for 3D man-made models, Pattern Recognition 45 (4) (2012) 1524–1530. doi:10.1016/j.patcog.2011.10.022.

[23] Y. Jin, Q. Wu, L. Liu, Unsupervised upright orientation of man-made models, Graphical Models 74 (4) (2012) 99–108. doi:10.1016/j.gmod.2012.03.007.

[24] M. Chaouch, A. Verroust-Blondet, Alignment of 3D models, Graphical Models 71 (2) (2009) 63–76. doi:10.1016/j.gmod.2008.12.006.

[25] A. Axenopoulos, G. Litos, P. Daras, 3D model retrieval using accurate pose estimation and view-based similarity, in: ACM International Conference on Multimedia Retrieval, 2011, pp. 41:1–41:8. doi:10.1145/1991996.1992037.

[26] M. Averkiou, V. G. Kim, N. J. Mitra, Autocorrelation Descriptor for Efficient Co-Alignment of 3D Shape Collections, Computer Graphics Forum 35 (1) (2016) 261–271. doi:10.1111/cgf.12723.

[27] X. Xie, J. Feng, Volumetric shape contexts for mesh cosegmentation, Computer Aided Geometric Design 43 (2016) 159–171. doi:10.1016/j.cagd.2016.02.006.

[28] M. Martinek, R. Grosso, Optimal rotation alignment of 3D objects using a GPU-based similarity function, Computers & Graphics 33 (3) (2009) 291–298. doi:10.1016/j.cag.2009.03.023.

[29] S. Althloothi, M. H. Mahoor, R. M. Voyles, A Robust Method for Rotation Estimation Using Spherical Harmonics Representation, IEEE Transactions on Image Processing 22 (6) (2013) 2306–2316. doi:10.1109/TIP.2013.2249083.

[30] J. Fehr, M. Reisert, H. Burkhardt, Fast and Accurate Rotation Estimation on the 2-Sphere without Correspondences, in: European Conference on Computer Vision, 2008, p. 239251. doi:10.1007/978-3-540-88688-4_18.

[31] P. Shilane, P. Min, M. Kazhdan, T. Funkhouser, The Princeton Shape Benchmark, in: Shape Modeling Applications, 2004, pp. 167–178. doi:10.1109/SMI.2004.1314504.

[32] Kalogerakis, Evangelos and Hertzmann, Aaron and Singh, Karan, Learning 3D mesh segmentation and labeling, ACM Transactions on Graphics 29 (4) (2010) 102:1–102:12. doi:10.1145/1833351.1778839.